ps11-1

Summary of guest speaker's presentations and learning experiences

Lecture by Dr. Drew Feiner

The invited lecturer was a CMU alumni and when he was pursuing his PhD degree, he researched into the academic analysis of blades of turbines. Then he started his own company, the Blade Diagnostics. Dr.Drew introduced his fantastic work at Blade Diagnostics and mainly talked about how Matlab helped his work in different ways. His company sell software or hardware to other companies to help them analyze their turbine blades or just for testing purposes. To tell the truth, I have seldom used Matlab before and Dr. Drew' s talk totally subverted my impression and expectation about Matlab. I have decided to pick up Matlab and explore something more about it. Then he also shared with us some obstacles he faced and came over during the development of his company as well as some of his previous experiences at CMU.

In all, Dr. Drew. showed us a great example of the successful transition from academic to the industry which might be of great importance to most of us. Also he introduced us the powerfulness of Matlab, making people like me cannot wait to have a try.

Lecture by Marc Zinck

The invited lecturer was also a CMU alumni graduated from the computer science department who also once worked in the Robotics Institute at CMU. As a current software engineer from Autodesk, Mr. Zinck talked about his experiences with a very common programming bug, the race condition, which is extremely malicious and also not that easy to inspect. In multithreads applications, when two or more threads can access and modify the shared data at the same time, race condition happens. Mr. Zinck gave us some examples of typical situations where race condition happens. And when it happens, all threads "race" to be executed first, we cannot know the execution sequence of these threads at all, thus making the result unpredictable. Then Mr. Zinck shared about how to prevent the race condition by locking the shared data. Once the share data are locked, the threads will be executed sequentially. When one thread is being executed, the other threads cannot access and modify the data until the data are unlocked. And finally he introduced us some ways to to lock the data, for example the mutex lock.

Actually just one day before this lecture I happened to made a race condition error in my homework of the computer system course. Through Mr. Zinck's lecture I had deeper and more comprehensive understanding over race condition. I will definitely be more careful to avoid this error in my future programs.